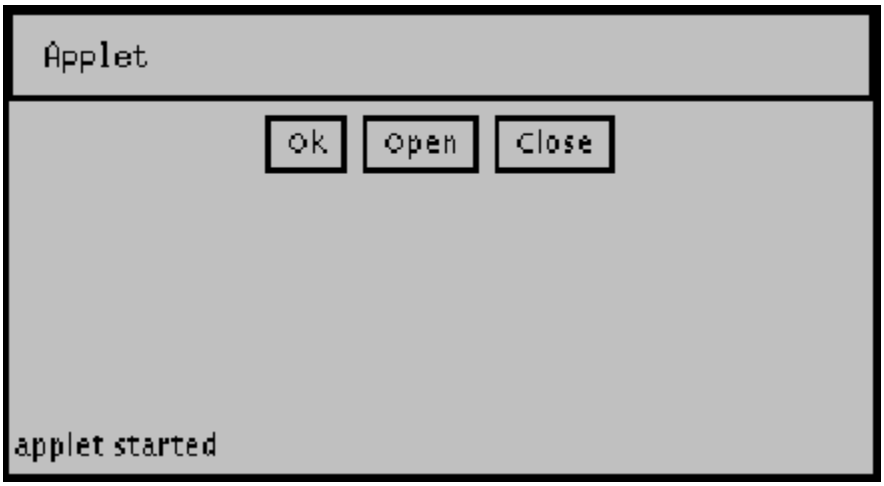
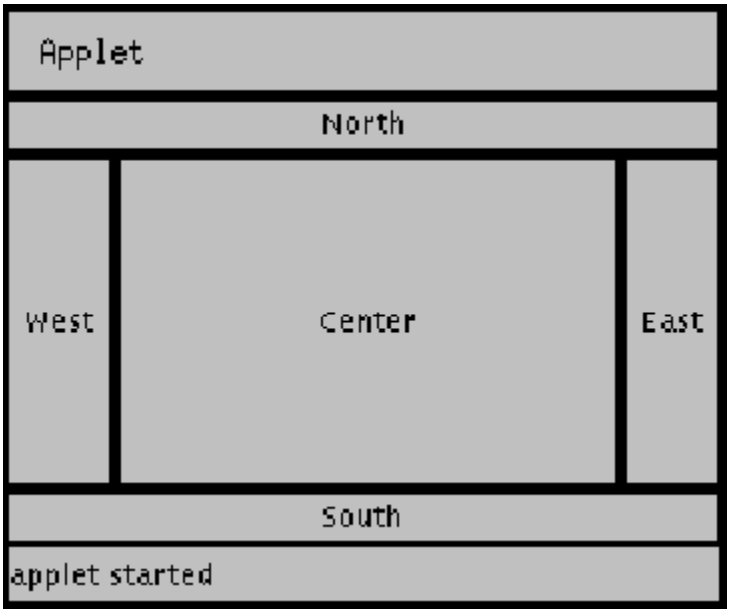
Recitation Guide Wednesday June 27, 2007

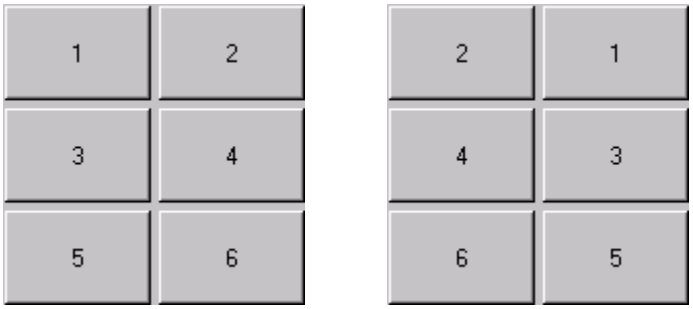
1. House keeping
    a. Basic Java Review Session
        i. Host(s): Rory and Kristin
        ii. When: Thursday June 28 @ 4:30pm
        iii. Where: CoC 102
    b. Quiz 2
        i. Average: 85%
        ii. Return quizzes and go over solution.
        iii. Rehash linked list methods:
            1. See TraversingTheLinkedList_part1-dsf.2.pdf
            2. See TraversingTheLinkedList_part2-dsf.pdf
        iv. Re-grades to Dawn and appealable to Professor Potts.
    c. Quiz 3
        i. Monday July 2 during class.
    d. Pair Issues
        i. Please report any pair issues to a TA or Professor Potts and if necessary we will help you find a new partner or allow you to work by yourself.
2. Intro to GUIs
    a. What is you need to import to use GUI components and ActionListeners
        i. import java.awt.*;
        import javax.swing.*;
        import java.awt.event.*;
        ii. You probably will not be required to memorize these for an exam or quiz.
    b. The containers: JFrame and JPanel
        i. Generally, JFrames hold JPanel objects and JPanel object in turn hold other objects such as JButtons and JTextFields
        ii. Different layouts for JFrame and JPanel



1. FlowLayout –Components are added starting at the left and go to right (default layout is always FlowLayout)



2. BorderLayout – Components are added based on a specified region. See picture above.



3. GridLayout – Components are added to a panel that is divided into a grid.
4. There are more but we usually focus on these in this course.

c. More components: JButton, JTextArea, JLabel, JTextField
   i. JButton – a button that can be pressed
   ii. JTextfield – a field where you can type and is only a single line high
   iii. JTextArea – a field where you can type and can be more than a single line high
   iv. JLabel – a label (in this course we typically add Pictures to the JLabel and then add the JLabel the GUI).
   v. As always there are more components but we tend to focus on these for this course.

d. What is the difference: Interfaces and Abstract class
   i. Abstract classes
      1. Can contain abstract methods.
      2. Can contain just regular methods.

3. Cannot be instantiated (i.e. cannot create a new instance)
4. Needs to be extended by a child class is order to use it (keyword extends).
5. Classes that extend the abstract class must override the abstract methods or Java will throw an error.

ii. Interfaces
1. Can only contain method headers followed by a semicolon (i.e. public void eat(); or public int count();)
2. Cannot be instantiated (i.e. cannot create a new instance)
3. Needs to be implemented by a class (keyword implements).
4. Classes that implement the interface must override all methods in an interface or Java will throw an error.

iii. Why do we want to use them?
1. Using interfaces basically streamlines production and makes sure everyone uses the same method signatures (method signatures include the method name, number and type of parameters and return type). A real world example would be the USB ports. Instead of having a bunch of different kind of ports, everyone just uses the same one.
2. Abstract classes are usually used when children of the class will different things with the same method. A common example of this is having an abstract Animal class and having an abstract method "speak," because each kind of Animal (or the children of the Animal class) will "speak" differently. Cats go "meow." Dogs go "bark." Etc.

e. ActionListeners and anonymous inner classes.

i. If you want something to actually occur when you press the button or do something to some other component, you need to place an ActionListener on the button or component.

ii. There several ways of creating your ActionListener. We usually recommend this way in this course:

```
JButton button = new JButton();
button.addActionListener(
                                    new ActionListener() {
                            public void actionPerformed(ActionEvent e) {
                                    //Some sort of action occurs in here
                            }
} );
```

1. Instead of declaring the ActionListener in a brand new file, we sort of declare it on the fly when we are adding it. The ActionListener, in this case, is called an anonymous inner class. It is called anonymous because we do not give it a name and inner because it is declared inside another class.

2. Remember that ActionListener is an interface (yeah I totally lied when I said we do not go over interfaces) so you need to override the abstract method actionPerformed.

iii. Also see http://java.sun.com/docs/books/tutorial/java/javaOO/innerclasses.html for more info on inner classes.

iv. Also see http://java.sun.com/docs/books/tutorial/uiswing/events/actionlistener.html for more info on ActionListeners.

f. The crazy thing about JLabels and Pictures

i. Since the classes in the java-sources were written specifically for this class, as a whole they are considered non-standard Java classes. The GUI components are java-standard so there are some incompatiabilities.

ii. The following lines are basically one of the only ways you can get a Picture onto a GUI:

```
JLabel pLabel = new JLabel();
Picture p = new Picture(FileChooser.getMediaPath("swan.jpg"));
Image image = p.getImage();
pLabel.setIcon(new ImageIcon(image));
```

g. Converting a String to a number

i. The following lines will help you convert a String number (such as "15" or "-9") from a JTextField or JTextArea to an int.

```
JTextField field = new JTextField("15");
String fieldInput = field.getText();
int i = Integer.parseInt(fieldInput);
```

3. Homework 7

a. Due Tuesday July 3 at 11:45pm with grace until Wednesday July 4 at 7:00am

b. Description: http://coweb.cc.gatech.edu/cs1316/633#hw7

c. Grading Criteria: http://coweb.cc.gatech.edu/cs1316/715